HOCHSCHULE DER MEDIEN

TERM PAPER

Course of studies: Library & Information Management

Stuttgart Media University

Faculty for Information and Communication

Publishing Linked Data - different approaches and tools

submitted by
Holger Büch

Advisor:	Prof. Magnus Pfeffer
Author:	Holger Büch Matriculation number: 26592 hb046@hdm-stuttgart.de
Period for editing:	8th of March, 2014 until 30th of May, 2014
Date of submission:	30th of May, 2014

Contents

List of abbreviations

API	Application Programming Interface
CMS	Content Management System
CSV	Comma Separated Values
FOAF	Friend of a Friend
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
LOD	Linked Open Data
LOV	Linked Open Vocabularies
OPM	Open Provenance Model
RDB	Relational Database
RDF	Resource Description Framework
RDFa	Resource Description Framework in Attributes
SPARQL	SPARQL Protocol and RDF Query Language
URI	Uniform Resource Identifier
VoID	Vocabulary of Interlinked Datasets
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	Extensible Markup Language

List of figures

1 Introduction

'I liked the idea that a piece of information is really defined only by what it's related to, and how it's related. There really is little else to meaning. The structure is everything.'[1]
(Tim Berners-Lee, 1999)

This quotation was stated by Berners-Lee at the end of the 20th century to describe the underlying idea of the World Wide Web (WWW) that he invented. And this idea is also the basic principle behind a ground-breaking method to publish information on the web. Seven years later this method was coined by Berners-Lee as Linked Data.[2] While the WWW started as a global information space of linked documents, there has been an increasing demand to link not only documents but also data in the last years.[3] The term 'Linked Data' is not used for *all* interconnected data, but only for data that complies with the following four rules proposed by Berners-Lee in a famous paper from 2006:[4]

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF*, SPARQL)
4. Include links to other URIs so that they can discover more things.

There is a long tradition in interconnecting data across different data-sources using network models or record linkages. But Linked Data, as proposed by Berners-Lee, has great advantages: data with globally unique Uniform Resource Identifiers (URIs) and modelled according to the Resource Description Framework (RDF) specification[5], recommended by the World Wide Web Consortium (W3C), offers a high degree of interoperability. Thus Linked Data can be easily used without being limited to specific applications or websites. The flexibility of RDF allows modelling the data according to ontologies that represent the concepts of the data. This makes Linked Data highly

[1] Berners-Lee (1999), p. 14.
[2] Cf. Berners-Lee (2006).
[3] Cf. Bizer, Heath, and Berners-Lee (2009), pp. 1-2.
[4] Berners-Lee (2006).
[5] Cf. W3C (2014).

reusable in a wide variety of contexts.

But how can data be published as Linked Data? What kind of actions have to be taken in order to transform data according to Berners-Lee's four principles? This term paper introduces basics procedures of this transformation process. The focus is on the theoretical process of publishing, aspects of technical realization of this process through different approaches and the description of a first attempt to put the publishing process into practice.

2 The steps to Linked Data

2.1 Variety of process models

The steps to Linked Data or Linked Open Data (LOD) (which includes appropriate free licensing and accessibility to data) are not standardized. Various guidelines exist. Some characterize the process as a life-cycle, other authors describe it as a linear sequence (Fig. 1-3). Hyland et al. proposed a model with seven steps (Fig. 1). As it is intended for governmental data, the major difference to other models is the absence of an exclusive step for linking the data to other datasets. Instead, it draws attention to the public announcement of the available data. Maybe the authors assume that if the community knew about the LOD, it would use and interconnect it with the LOD Cloud and therefore linking is not as important as the other activities.

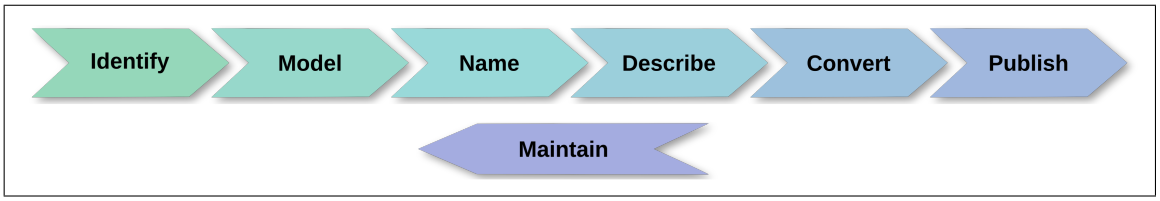

Figure 1: Steps for publishing Linked Open Data by governments.[6]

The model by Auer et al. with its eight steps is more detailed and focuses on the technical aspects of the publishing process (Fig. 2). It shows the stages of the LOD life-cycle that can be supported by the LOD2 Stack, which is an integrated distribution of software to support LOD.[7] The form of a life-cycle might not be the best representation for these different steps of the work process, because the given order of the steps in the cycle seems to be somehow arbitrary. For example 'authoring' might be better described as a continuous effort through the whole process than as a single task at a specific time. Despite the improper representation, this model provides a helpful overview on software required for the process: it illustrates the tasks which represent different fields for applications. Every single step can be related to specific software-tools.

[6] Illustration by the author, based on Hyland and Villazón-Terrazas (2011).
[7] Cf. Auer et al. (2012), pp. 1-2.

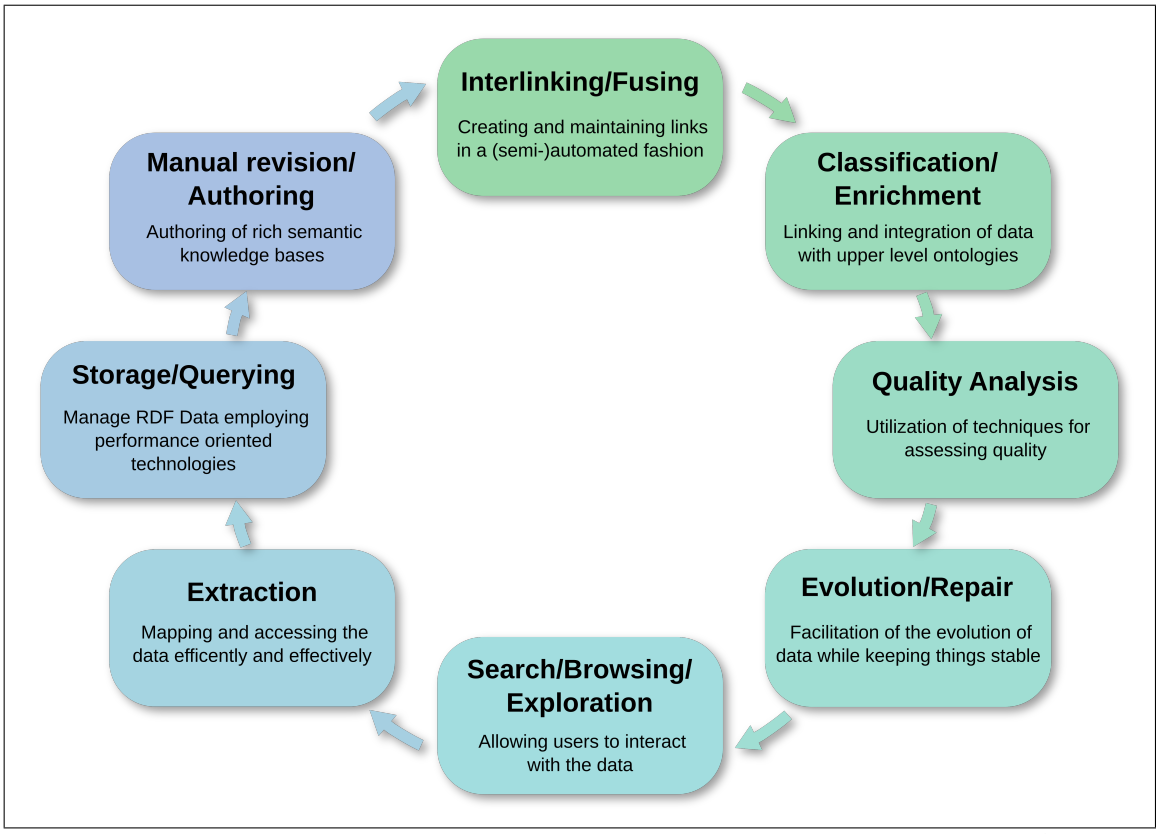

Figure 2: Stages of the Linked Data life-cycle supported by the LOD2 Stack.[8]

The process description chosen for this term paper is the model by Villazón-Terrazas et al. (Fig. 3). Based on an earlier model,[9] it has been extended by the step 'linking', which was previously included in the step 'generation'. The model gives a good overview about the general aspects of the publishing procedure and reflects the basic work-flow to realize the procedure technically. That is the reason why this model is explained in details here.

2.2 Activities for publishing Linked Data

The model by Villazón-Terrazas et al. describes activities for publishing Linked Data. Its intended use is to serve as a methodological guideline for publishing governmental data, but the general model is supposed to apply to almost every other data, too. Every single activity can be implemented through a couple of tasks and can be supported by the use of software-tools.

[8] Illustration by the author, based on Auer et al. (2012), pp. 2-3.
[9] Cf. Villazón-Terrazas, Vilches-Blázquez, et al. (2011), p. 30.

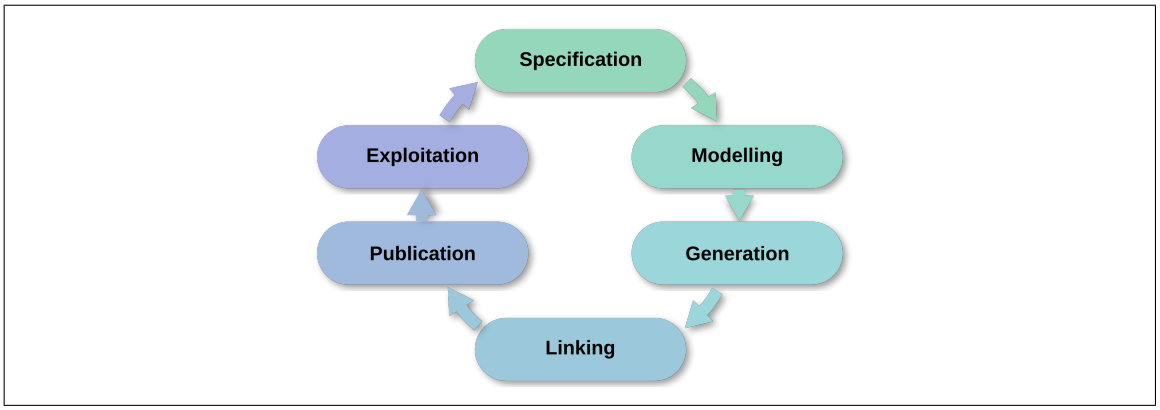

Figure 3: Main activities for publishing Linked Data.[10]

During the activity **specification**, basically three tasks are required. First, the data that should be published has to be identified and analysed. Which data sources are used? In which form is the data available? What is the structure and meaning of the data? After finding answers to questions like that, the structure of the URIs can be designed as a second task. In the third task legal issues like licensing can be addressed.[11]

Modelling is the second activity. Therefore, an ontology has to be created or chosen to represent the information of the domain of the data as a set of hierarchical concepts. It is advised to reuse existing vocabularies as much as possible. This speeds up the activity and facilitates the use of the data by third persons.[12] Repositories like Linked Open Vocabularies (LOV) host hundreds of vocabularies and make them searchable for domains, classes or properties.[13] A visualization of the history and references indicates the importance and maintenance of the vocabularies (Fig. 4) and thus helps to make the appropriate choice.

The purpose of the activity **generation** is the transformation of the data to RDF according to the Linked Data principles. The previously selected ontology is used to shape the data.[14]

[10] Illustration by the author, based on Villazón-Terrazas, Vila-Suero, et al. (2012), p. 2.
[11] Cf. Villazón-Terrazas, Vilches-Blázquez, et al. (2011), pp. 30-33.
[12] Cf. ibid., pp. 33-34.
[13] Cf. Vandenbussche and Vatant (n.d.[a]).
[14] Cf. Villazón-Terrazas, Vilches-Blázquez, et al. (2011), p. 34.

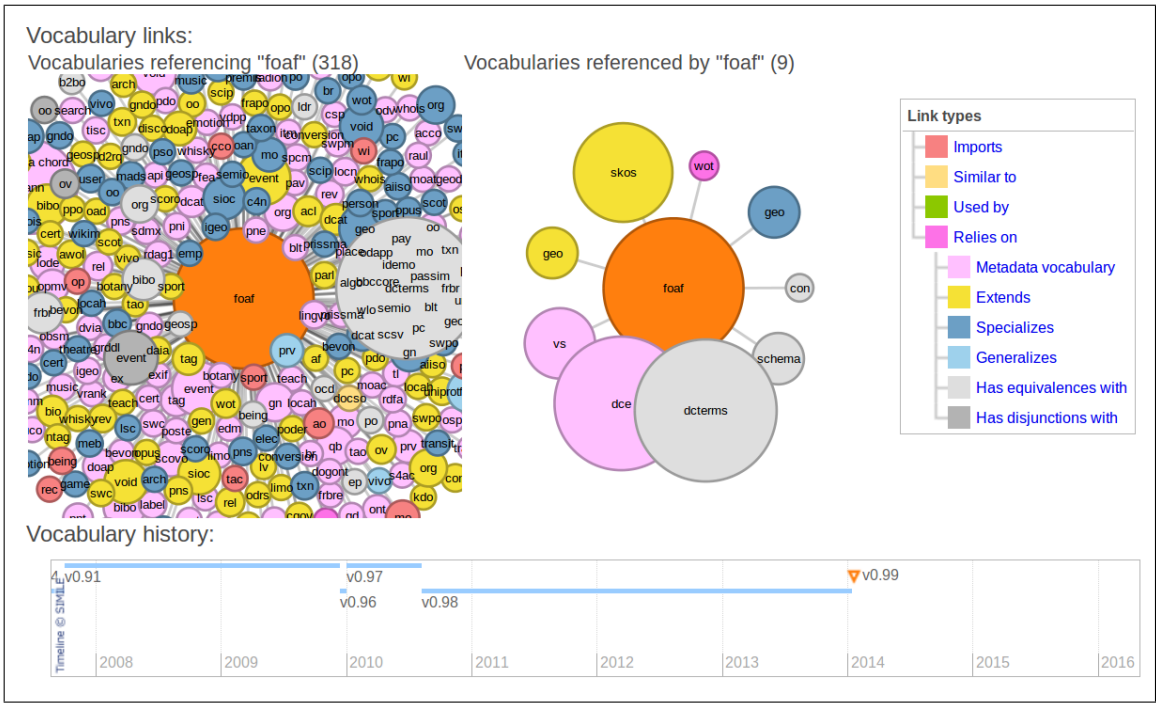

Figure 4: Visualization of the history and references of the FOAF vocabulary.[15]

Linking the data to other datasets is the fourth principle of Linked Data (s. Chap. 1) and the fourth activity in this model. Available datasets suitable for linking have to be identified. Datahub[16] and other repositories are places to look for other LOD datasets. Datasets should be selected carefully, because accounting the number of broken links on Datahub lots of their sets seem to be not persisting. After the relationships between items of the own dataset and other datasets on the web have been identified, the linking can be done by assigning the URIs to properties like for example ‘sameAs’ or ‘seeAlso’.[17]

After that, the Linked Data is ready for **publication**. The aim of this activity is to store the data in a database for triples which is connected to the web, so it can be exported (usually as RDF). It is desirable to provide access to the data for humans and computer programs. For humans a Linked Data front-end software should be implemented. While for computers programs a SPARQL Protocol and RDF Query Language (SPARQL) endpoint can be considered as standard.[18] Beside the data itself the meta-data description of the Linked Data should be published. For this purpose

[15] Screenshot by the author, see Vandenbussche and Vatant (n.d.[b]).
[16] See Open Knowledge Foundation (n.d.).
[17] Cf. Villazón-Terrazas, Vilches-Blázquez, et al. (2011), pp. 36-37.
[18] Cf. ibid., p. 37.

special vocabularies like the Vocabulary of Interlinked Datasets (VoID) can be used.[19] With models like the Open Provenance Model (OPM) the provenance information of the data can be represented and provided for access.[20] Beside providing access to the data, the data should also be promoted. This can be done by creating `sitemap.xml` files as a map of the available data and submitting these files to the major search engines, so internet-users can find the data through their usual access-points. Also the description of this new source of Linked Data should be published over repositories like the already mentioned Datahub.[21]

The last activity in this model is the **exploitation** of data. Especially for publicly funded Linked Data efforts it's important to show the value of these efforts to the citizen. Most of the work and value of Linked Data lies under the surface, so the benefits might be hard to recognize for non-technical individuals. Therefore applications should be implemented that integrate data from different sources and provide access to rich graphical user interfaces which demonstrate the power of Linked Data and give additional value to the final users.[22]

These six activities only give an overview of the various tasks that have to be addressed to publish Linked Data. Taking a closer lock, more decisions have to be made and a lot more parameters have to be considered. An examination from the basic steps to the realization of the presented work-flow can illustrate these difficulties.

[19] Cf. Alexander et al. (2011).
[20] Cf. Moreau et al. (2011).
[21] Cf. Villazón-Terrazas, Vilches-Blázquez, et al. (2011), pp. 37-38.
[22] Cf. ibid., pp. 38-39.

3 Realization of the work-flow

3.1 A multifaceted model of the work-flow

The process model published by Heath and Bizer (Fig. 5) shows different work-flows and options for publishing Linked Data. Of course it is limited and intended to cover only the most common patters.[23] It's arguable, if publishing structured data and text (as an example for unstructured data) is really the most common usage for Linked Data: maybe there should also be a work-flow for semi-structured data. But the publishing process of semi-structured data can probably be described pretty well by a mixture of the activities used for structured data and text. Therefore this inaccuracy isn't very significant and can be disregarded in this context.

In the following it will be explained how these work-flow options by Heath and Bizer can be mapped to the activities for publishing Linked Data by Villazón-Terrazas (s. Chap. 2.2).

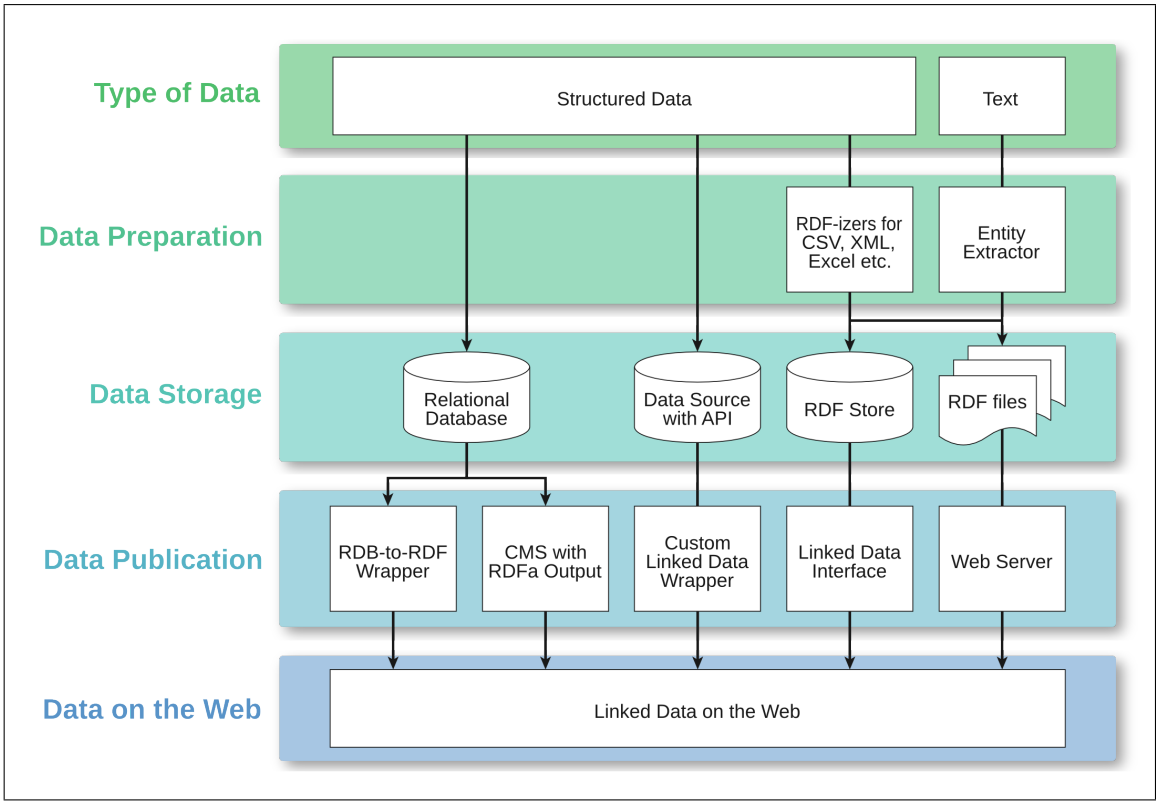

Figure 5: Linked Data publishing options and work-flows.[24]

[23] Heath and Bizer (2011), chap. 5.1.
[24] Illustration by the author, based on ibid., chap. 5.1

1. **Specification**. First we have to identify in which form and format the data is currently available. Corresponding to figure 5 we have four options: depending on whether we are dealing with structured data stored in a Relational Database (RDB), structured data somehow made available via an Application Programming Interface (API), structured data in format likes Comma Separated Values (CSV) or as text we have different possibilities.

2. **Modelling**. The modelling activity is not explicitly expressed in the chart, but it has to be completed before generating the RDF.

3. **Generation**. We have two options to generate the RDF out of a RDB: using Wrapper software or, if the RDB is part of an Content Management System (CMS), we might be able to use build-in solutions to embed the data as Resource Description Framework in Attributes (RDFa) directly into generated Web documents.
   To transform the data provided by an API, most likely a custom made or highly adapted Wrapper software is necessary.
   CSV-files or other files with structured data can be converted by using RDF-izer software, that can be configured to generate triples by following custom rules.
   Obviously the most difficult step is to convert unstructured data like text to RDF. Complex software, a so called Entity Extractor, is used to automate this process.

4. **Linking**. Adding links to other datasets on the Web is also not explicitly expressed by the chart, but it can be performed during the transformation to RDF. At the latest this can be done during the final data publication process.

5. **Publication**. The data produced by Wrappers (no matter if the origin is an RDB or an API) is usually published by providing an SPARQL interface. The RDFa produced by CMSs might be delivered as Hypertext Markup Language (HTML)-Website or Extensible Markup Language (XML)-Feed.
   Data stored as RDF can be published via a usual Web Server if static files are used. If the data-hosting is done with an RDF Store, the data can be accessed via SPARQL-endpoint.
   As an SPARQL-endpoint is good for accessing the data with programs, sometimes access via URI is desired. If the Wrapper or RDF Store doesn't provide this feature, an interface software can be used to dereference URI-requests into SPARQL-queries and return the data in different formats.

6. **Exploitation**. The application to use and promote the Linked Data is out of scope of Heath's and Bizer's model.

The presentation of this process model gave a good impression on the most basic options to handle the publishing of Linked Data. Even when simplified, it brings the work-flow's complexity to mind. More detailed information about the function of the needed tools seems to be useful to put the theory into practice. Hereby only the tools are explained, that are of value for the experiential report [s. Chap. 4].

3.2 Common categories of tools

3.2.1 Triple store

The term triple store is synonymic to the term RDF store. It is a software solution for storing and managing data entities in form of an ordered list of three elements, the so called triples. The common function of these three elements is to relate a subject to an object by using a predicate. A natural language example would be: 'Plato was born in Athens'. Another example would be: 'Athens is located in Greece'. It's possible to derive the semantic meaning from these two examples and generate a third statement: 'Plato was born in Greece'. It's very easy for a human to draw this conclusion, but more difficult for computers. Tasks like this can be implemented into software by using the RDF specification. As computers can access and process billions of triples, they are able to solve much more complex semantic tasks like the examples given here. But the huge amount of triples have to be stored and managed. That's the purpose of triple stores.

The two main functions of a triple store are the storing of the triples in a highly specialized database for fast access and to provide an interface for manipulating and querying the data usually via SPARQL queries. The management of access restrictions is also possible.

Some major products offer support for integrating a wide variety of heterogeneous data. For example OpenLink Virtuoso can store data not only in RDF but also as relational data, object-relational data and other forms. It offers additional features like a Web Application Server and a Linked Data Server based on Hypertext Transfer Protocol (HTTP). Virtuoso exists in an Open Source edition and in a commercial edition with additional features.[25] A 100% Open Source solution is Bigdata, a triple store that claims to be optimized for high scalability and performance with large datasets.[26]

Considering, that the area of the 'Web of data' is just at its beginning, the major

[25] Cf. OpenLink Software (2014).
[26] Cf. Systap (2014).

challenge to prepare triple stores for the future is performance at large scales. That's why there is an increasing interest in developing distributed data storage solutions which can work in server clusters scattered in the cloud and handle massive amounts of triples.[27]

3.2.2 RDB Wrapper

Specialized triple stores are undisputed the highest performing storage solutions for large RDF datasets. But there still are many cases, where the data that should be exposed to the Web as Linked Data is stored in RDBs. Most likely, because it is used for an important internal application which depends on it. To be able to publish this kind of data a so called Wrapper software is needed. The Wrapper is embedded as an extra layer between the user and the non-RDF database. It can provide access points like an SPARQL endpoint for computers, HTML documents to show the data to humans or RDF files for RDF browsers. To accomplish this, the Wrapper transfers the query for RDF data in a query to the RDB. This is possible with a predefined mapping description that tells the Wrapper which RDF terms are related to which data in the database. Then the Wrapper receives the answer of its query from the RDB, transforms it into RDF using the mapping description again and makes it available it in the desired form.[28]

A common example for a Wrapper is the D2R Server. It's Open Source and comes in a bundle with other tools under the name D2RQ Platform. This set of tools supports the mapping process and adds additional features like the ability to dump the data from the RDB into a single RDF file, that could then be loaded into a triple store. This software solution was developed by Bizer, is now maintained by Cyganiak and supported by the Freie Universität Berlin and other organisations.[29]

Another example is the CMS Drupal. It runs with a RDB in the back-end, but its core includes a RDF module to output data as RDFa. It also supports the mapping of data to imported ontologies. Developers are trying to improve the functionality by developing extensions, for example to support the output into formats like RDF or Turtle.[30]

Regarding the current popularity for RDBs, it is highly probable that a huge amount of data that has to be published as RDF will be kept maintained as relational data

[27] Cf. Mika and Tummarello (2008), pp. 82 and 87.
[28] Cf. Heath and Bizer (2011), chap. 5.2.4.
[29] Cf. Cyganiak (n.d.).
[30] Cf. Corlosquet et al. (2009), pp. 763-766.

in the mid-future. Therefore Wrappers are considered as an important part of the infrastructure for the Semantic Web.[31]

3.2.3 RDF-izer

This category of tools has the similar purpose as a Wrapper: convert data from other formats to RDF. But different to a Wrapper, a RDF-izer is not intended to transform constantly changing data live. RDF-izers are usually used if the data source is a given static dataset.[32] Because the transformation of this data is usually easier and doesn't depend on performance very much, RDF-izers can handle more diverse data sources than Wrappers. Common source formats are CSV or Excel, but a huge list of converters for many other formats is available in the W3C Wiki.[33]

A powerful tool to analyse data and exchange it between different data-formats is OpenRefine (formerly known as google-refine). The extension RDF Refine enables OpenRefine to transform data to RDF and provides additional features like searching for related datasets on the Web.[34]

3.2.4 Interlinker

To create actually Linked Data, the RDF has to be linked with other datasets in the Web. This allows to follow the RDF links of a dataset at run-time to discover new data sources and information. And it helps to integrate this data into other systems, because it is easier to understand a dataset if links to commonly used vocabularies are embedded.[35]

To be able to add links to our own data, we have to know about the other datasets and we have to understand the foreign data to be able to find the relation to our own data. For example if our own dataset has a field `Name:Plato` and shall be linked with another dataset, where the name is spelled `Platon` it might be difficult for computers to match the entities correctly. Fortunately, software from the category Interlinker can support the user in this process. For example the Open Source tool Silk Link Discovery Framework has heuristic algorithms on board to compare datasets based on their similarity. It has matchers for strings, numbers and geographical values, and

[31] Cf. Bizer and Cyganiak (2006), p. 2.
[32] Cf. Heath and Bizer (2011), chap. 5.1.1.
[33] Cf. W3C Wiki contributors (2014).
[34] Cf. Linked Data Research Centre at the National University of Ireland (n.d.).
[35] Cf. Heath and Bizer (2011), chap. 2.5.3.

a build-in preprocessor to transform the data values before the matching process, for example by using regular expressions.[36]

3.3 Integrated Linked Data solutions

It is still complex to handle the different tools that are needed for the publishing work-flow of Linked Data. Discovering and evaluating the software, arranging the different requirements for tools, installing and configuring the selected solutions requires a lot of experience and knowledge. Then the chosen products have to be interconnected and tested, to find out if everything is working together as intended. One of the smaller problems that also have to be considered is familiarizing the user with different interfaces and terms the tools might use. So setting up the whole process with the different tools available requires both a lot of time and resources. This is a huge barrier for smaller institutions. Some teams are already working to break down this barrier by developing integrated Linked Data solutions.

Integrated Linked Data solutions try to combine the multiple tools as a set of pre-configured components in a single environment and with a unified user interface. The procedure of installing and connecting the different components is hidden in a single installation routine. In the best case, the user would not notice that he is installing and using multiple tools. The user should get the impression of using one single piece of software. Furthermore, experienced users are still able to use, tune or replace single components. They just have to leave the unified user interface for this purpose.

Considering the difficulties with implementing such integrated solutions, it's no surprise that there is no finished product available currently. All three solutions covered in the following chapters have the status of ongoing research projects.

3.3.1 LOD2 Stack

The solution LOD2 Stack is an initiative of researchers at the University of Leipzig and part of the LOD2 research project funded by the EU. It consists of 20 different components, 16 to be installed locally and 4 online components.[37] The LOD2 Stack is intended to help managing the whole life-cycle of Linked Data. Thereby the developers want to cover a broad set of use cases and requirements. To reach this ambitious goal, LOD2 Stack is based upon three pillars defined by the researcher team:[38]

[36] Cf. Heath and Bizer (2011), chap. 4.5.4.
[37] Cf. LOD2 Wiki contributors (2014).
[38] Cf. Auer et al. (2012), pp. 1-2.

1. Using the Debian package system to ease the deployment of the different components.

2. Using a central SPARQL endpoint with standardized vocabularies as a central knowledge repository. All the other components manipulate and read the data through this access point.

3. Providing a central user interface to manage the different components.

Unfortunately, the researchers are not publishing much information about the status of this project. The website `lod2.eu` is not accessible and the blog of at `stack.lod2.eu` offers only minimal information. This is probably because the funding of the project seems to have ended in 2013. Nevertheless `wiki.lod2.eu` is a good resource about the background of the project and seems to be updated regularly, just like the public repository for the LOD2 Stack Debian packages. A survey about the usage of LOD2 Stack was done 2013, indicating a general interest in this kind of integrated solution.[39]

3.3.2 DataLift

DataLift is another research project for building an integrated Linked Data solution. Its aim is to become a 'catalyser for the Web of data' by providing support for selecting good ontologies, converting data into RDF, publishing the data on the web (by providing access right management) and interlinking the data with external datasets. It's funded by the French national research agency, is lead by the French National Research Institute on Computer Science and Control and has partners from other institutions and the industry.[40]

DataLift doesn't include as diverse components as the LOD2 Stack. Therefore it intends to have fewer features out of the box, but its architecture allows extensions by third party modules. The basic work-flow for publishing data looks complex (Fig. 6), but the software does most of the steps and the user is guided through the whole process with an easy to understand interface.

First, the user has to submit a source of structured data to the system. Seven different formats and sources are currently supported, including CSV, RDF formats, RDB and SPARQL endpoints. This data is then converted to raw RDF by the system, without taking into account vocabularies, links or name-spaces yet. The reason for

[39] Cf. LOD2 Wiki contributors (2013).
[40] Cf. Scharffe et al. (2012), pp. 25-27.

this simple conversion is that from now on DataLift and its components only have to handle this single format. Then the user is asked to input a set of vocabulary terms that describe the data. The terms can be mapped to the raw RDF and are then converted to a properly formatted RDF. Then this dataset can be enhanced by replacing names with the URI of another dataset. For example 'Plato' can be replaced by 'http://dbpedia.org/resource/Plato'. Then the data can be published on an SPARQL endpoint. As a last step, the data can be linked to other datasets on the Web. This module in DataLift basically generates a script for the Silk Interlinker by asking the user some simple questions.[41]

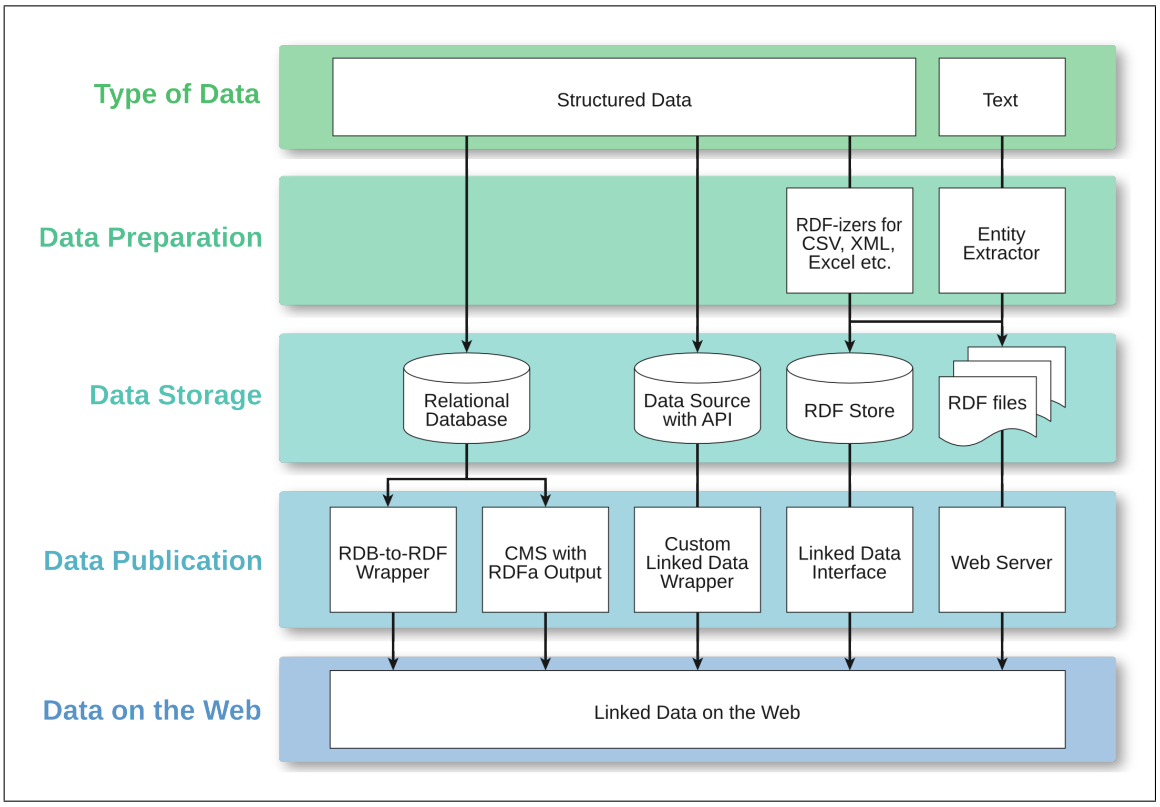

Figure 6: Datalift data workflow.[42]

The DataLift-Team shows more effort in informing about their work. A lot of papers are published on their website `datalift.org`. Unfortunately the usage documentation is not as detailed as their research information. That's an obstacle for early adopters, but comprehensible considering the early state of the development.

Compared to the LOD2 Stack, the concept of DataLift seems to be more promising to me. The objective of both products to develop an integrated solution for a broad

[41] Cf. Scharffe et al. (2012), p. 28.
[42] Illustration by the author, based on Cf. ibid.

application area is ambitious. DataLift's focus on the basic tasks might increase the change of success and it makes it easier to use. That's why DataLift was chosen as Software for the experiential report of this term paper.

3.3.3 Linked Media Framework

The concept of Linked Media Framework differs a little from the two solutions above. The software is intended to be used in many cases too, but there is a distinct focus on features used for publishing media data. This is no surprise: the New Media Lab in Salzburg started developing this application. The focus on media is manifesting itself in the integration of specialized features like entity extraction using Apache Stanbol. Text classification and a semantic search are other characteristics of Linked Media Framework. In addition, a client library is provided to ease the development of applications that connect the framework and the software is well documented.[43]

[43] Cf. LMF contributors (n.d.).

4 Experiential report: first try to publish Linked Data

4.1 Initial situation and installation

To get an idea about the time and effort needed for publishing Linked Data, a simple test seemed to be useful. Because of the importance of integrated Linked data solutions, the software DataLift (s. Chap. 3.3.2) was selected. The goal was to publish a dataset as Linked Data and link it to another dataset on the Web.

The test was performed on a virtual machine. Ubuntu Server was used as operating system and Apache Tomcat as Web server with support for Java. Then the installation of the DataLift components began. The only used installation guide was the documentation on the DataLift Wiki.[44] Surprisingly not all components needed were bundled in the DataLift download. OpenRDF Sesame as a RDF framework had to be downloaded and installed separately. According to the documentation, this is only needed for the server deployment. In a standalone deployment Sesame is included. The compiling of the DataLift source itself by using the Ant build tool went without problems.

Several compiled files had to be moved to different locations to get DataLift up and running. Unfortunately, the documentation isn't specific enough which files are exactly needed. Therefore some files where forgotten and that interrupted the later work-flow.

First it is required to create RDF repositories in Sesame to do the DataLift configuration.This is not difficult with Sesame's Web interface, but every step that has to be done outside the DataLift environment is against the goal of an integrated solution. The repositories were created according to the documentation. So there was no need to change the DataLift configuration file and the connection to the DataLift Web interface could be established.

[44] Cf. DataLift Wiki contributors (2013).

4.2 Performing the work-flow

4.2.1 Importing and converting to RDF

A list of most expensive paintings was extracted from the Wikipedia[45] and saved as a CSV file as data source. Importing this file was quite easy. Only a few parameters had to be filled out (Fig. 7). After the import, the data had to be converted to raw RDF before it was possible to map it with an ontology.

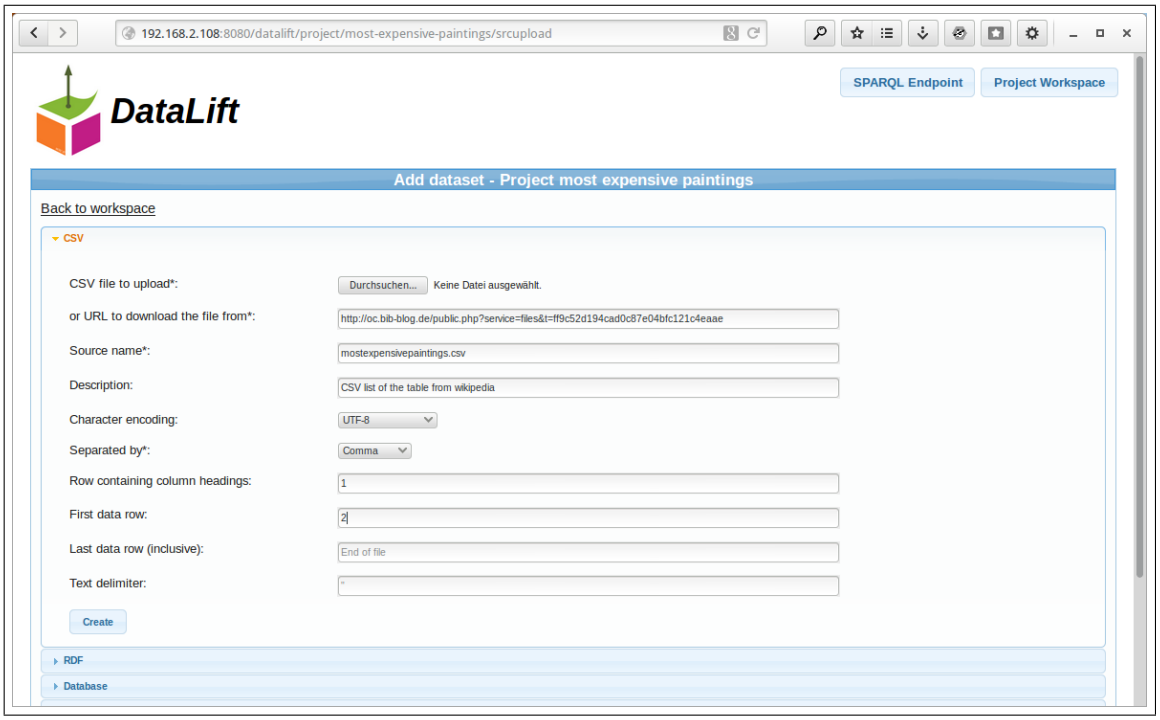

Figure 7: Importing CSV in DataLift.[46]

4.2.2 Mapping the data to the ontology

The mapping process was more difficult. An ontology can be loaded by entering an URL., Unfortunately, it didn't work in this test for famous ontologies like Dublin Core. However, the Friend of a Friend ontology could be used. The mapping process itself is well guided. The user can select fields of his own data and choose the corresponding field of the ontology. Before the converting process can be started, a graphical representation of the new structure is presented to the user (Fig. 8).

[45] See Wikipedia contributors (2014).
[46] Screenshot by the author.

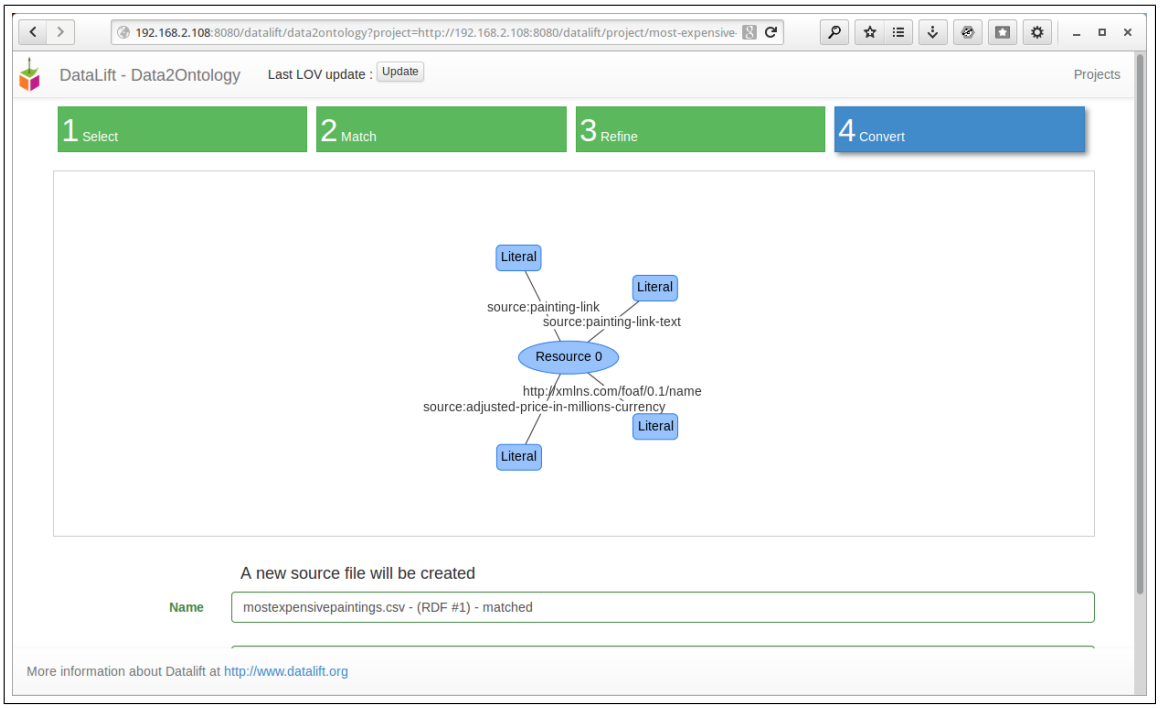

Figure 8: Mapping in DataLift.[47]

4.2.3 Interlinking and publishing

The interlinking process was the most difficult one. DataLift allows linking with both: local and remote datasets. The attempt to interlink with remote datasets failed in this test: it seemed that DataLift couldn't connect to these sources. Therefore the dataset had to be downloaded and imported to DataLift. Here a dataset from Amsterdam Museum[48] was chosen, after an import of the DBpedia Persons dataset failed. Probably because of its size.

The interlinking module based on the Silk interlinker offers a lot of features for string manipulation and similarity checking. Again, the interface guides the user well through the options (Fig. 9). Unfortunately, the Silk script generated by DataLift contained an error: the command for importing the linked data into Sesame was wrong (maybe outdated). But even after correcting the command, DataLift wasn't able to create the dataset in Sesame. However, the dataset could be saved locally and imported into Sesame manually. After this procedure, all three datasets (the list from Wikipedia, the dataset from Amsterdam Museum and a dataset containing the links) was accessible through DataLift's SPARQL endpoint.

[47] Screenshot by the author.
[48] See Boer et al. (2014).

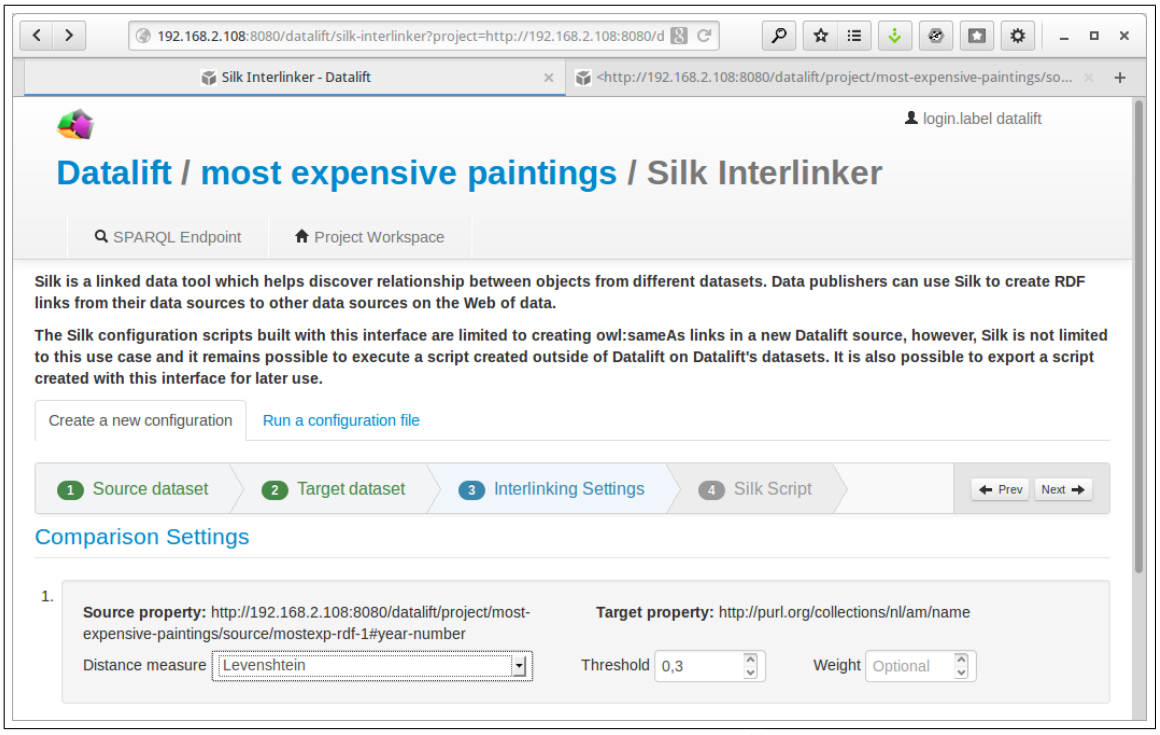

Figure 9: Linking in DataLift.[49]

4.3 Results

The installation was more complicated than expected. Software from different locations had to be installed and many things had to be done manually. An installation routine that supports the user in this process is missing and the given documentation is not detailed enough.

The well designed user interface of DataLift is astonishing. It guides the user through the difficult process of publishing Linked Data very well, while hiding unnecessary technical details. It can be easily understood with basic knowledge about Linked Data and looks appealing.

During the publishing process, lots of problems occurred. No information about these errors was given to the user through the interface, monitoring the log files was necessary. It was possible to do workarounds with time, effort and knowledge. But the goal of providing an 'install and use' solution has not been reached yet. Again it has to be noticed that DataLift is a research project in beta stadium, where problems like this have to be expected.

[49] Screenshot by the author.

5 Conclusion

The possibilities and options for publishing Linked Data is astonishing. Nearly any kind of data can be converted into Linked Data by using the right tools, and it can be provided in multiple forms accessible for humans and computer programs. The usefulness of Linked Data is undisputed.

Nevertheless, the publishing of Linked Data and the development of applications that make use of it is mostly limited to experts and institutions at the moment. The main reasons seem to be the required background knowledge and the confusing variety of tools which have to be discovered, installed and interconnected.

It is necessary to make the process of data publication a lot easier to turn the Web of Linked Data into a success similar to the Web of documents. One option to do this is creating integrated Linked Data solutions like DataLift. In my opinion, the necessity of such solutions can't be rated high enough. Stability and easy usage is at this stage more important than features. That's why I prefer the concept of DataLift over LOD2 Stack. An additional option is to limit the use cases for such software and develop integrated solution for a narrow field instead of trying to build a universal tool. This probably will increase the stability and easy usage.

But building integrated solutions is not the only way. Another possibility is to create a knowledge base of publishing Linked Data for different use cases and with different tools. In my opinion, a collection of recipes with detailed instructions about publishing Linked Data and a simple tool that helps the user to decide which recipes he should consider could also help to boost the Web of data.

Beside the two mentioned options, other options have to be developed and other actions have to be taken to break down the barrier for publishing Linked Data. Only if less time and knowledge is required for the publishing process (and for developing applications, too) the Web of data could become a grassroots-driven movement following Tim Berners-Lee intention, which is desirable.

References

Alexander, Keith et al. (2011): Describing Linked Datasets with the VoID Vocabulary. URL: http://www.w3.org/TR/void/ (visited on 05/19/2014).

Auer, Sören et al. (2012): Managing the Life - Cycle of Linked Data with the LOD2 Stack. In: International Semantic Web Conference (2), pp. 1–16. URL: http://iswc2012.semanticweb.org/sites/default/files/76500001.pdf (visited on 04/23/2014).

Berners-Lee, Tim (1999): Weaving the Web: the origins and future of the World Wide Web. London: Orion Business.

Berners-Lee, Tim (2006): Linked Data - Design Issues. URL: http://www.w3.org/DesignIssues/LinkedData.html (visited on 10/05/2014).

Bizer, Christian and Richard Cyganiak (2006): D2R Server - Publishing Relational Databases on the Semantic Web. URL: http://wifo5-03.informatik.uni-mannheim.de/bizer/pub/Bizer-Cyganiak-D2R-Server-ISWC2006.pdf (visited on 04/29/2014).

Bizer, Christian, Tom Heath, and Tim Berners-Lee (2009): Linked Data - The Story So Far. In: International Journal on Semantic Web and Information Systems 5.3, pp. 1–22. DOI: 10.4018/jswis.2009081901.

Boer, Victor de et al. (2014): Amsterdam Museum as Linked Open Data in the Europeana Data Model. URL: http://datahub.io/dataset/amsterdam-museum-as-edm-lod (visited on 04/17/2014).

Corlosquet, Stéphane et al. (2009): Produce and Consume Linked Data with Drupal. In: The Semantic Web - ISWC 2009. Springer, pp. 763–778.

Cyganiak, Richard (n.d.): D2RQ - Accessing Relational Databases as Virtual RDF Graphs. URL: http://d2rq.org/d2r-server (visited on 04/18/2014).

DataLift Wiki contributors (2013): DataLift - Platform installation. URL: http://datalift.org/wiki/index.php/Platform%5C_installation%5C_(english) (visited on 05/12/2014).

Heath, Tom and Christian Bizer (2011): Linked Data: Evolving the Web into a Global Data Space. URL: http://linkeddatabook.com/book (visited on 04/27/2014).

Hyland, Bernadette and Boris Villazón-Terrazas (2011): Cookbook for Open Government Linked Data. URL: http://www.w3.org/2011/gld/wiki/Linked%5C_Data%5C_Cookbook (visited on 04/05/2014).

Linked Data Research Centre at the National University of Ireland (n.d.): RDF Refine - Documentation. URL: http://refine.deri.ie/docs (visited on 05/17/2014).

LMF contributors (n.d.): Link Media Framework. URL: https://code.google.com/p/lmf/ (visited on 05/12/2014).

LOD2 Wiki contributors (2013): LOD2 Demonstrator-Survey 2013 - Introduction. URL: http://wiki.lod2.eu/display/LOD2DS13/Introduction (visited on 05/17/2014).

LOD2 Wiki contributors (2014): LOD2 Stack Components. URL: http://wiki.lod2.eu/display/LOD2DOC/LOD2+Stack+Components (visited on 05/17/2014).

Mika, Peter and Giovanni Tummarello (2008): Web semantics in the clouds. In: IEEE Intelligent Systems 23.5, pp. 82–87.

Moreau, Luc et al. (2011): The Open Provenance Model core specification (v1.1). In: Future Generation Computer Systems 27.6, pp. 743–756. URL: http://eprints.soton.ac.uk/271449/ (visited on 05/17/2014).

Open Knowledge Foundation (n.d.): Datahub. URL: http://datahub.io (visited on 04/17/2014).

OpenLink Software (2014): Virtuoso Universal Server - Features. URL: http://virtuoso.openlinksw.com/ (visited on 05/15/2014).

Scharffe, François et al. (2012): Enabling linked-data publication with the datalift platform. In: Proc. AAAI Workshop on Semantic Cities. URL: http://www.aaai.org/ocs/index.php/WS/AAAIW12/paper/viewFile/5349/5678 (visited on 04/14/2014).

Systap (2014): Bigdata. URL: http://www.systap.com/bigdata (visited on 05/15/2014).

Vandenbussche, Pierre-Yves and Bernard Vatant (n.d.[a]): Linked Open Vocabularies - About. URL: http://lov.okfn.org/dataset/lov/about/ (visited on 05/23/2014).

Vandenbussche, Pierre-Yves and Bernard Vatant (n.d.[b]): Linked Open Vocabularies - Friend of a Friend vocabulary. URL: http://lov.okfn.org/dataset/lov/details/vocabulary%5C_foaf.html (visited on 05/22/2014).

Villazón-Terrazas, Boris, Daniel Vila-Suero, et al. (2012): Publishing Linked Data - There is no One-Size-Fits-All Formula. In: Proc. of the First European Data Forum. Copenhagen. URL: http://ceur-ws.org/Vol-877/poster5.pdf (visited on 04/25/2014).

Villazón-Terrazas, Boris, Luis Vilches-Blázquez, et al. (2011): Methodological guidelines for publishing government linked data. In: Linking Government Data. New York: Springer, pp. 27–49.

W3C (2014): RDF 1.1 Concepts and Abstract Syntax. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/ (visited on 04/23/2014).

W3C Wiki contributors (2014): ConverterToRDF. In: W3C - Wiki. URL: http://www.w3.org/wiki/ConverterToRdf (visited on 05/20/2014).

Wikipedia contributors (2014): List of most expensive paintings. In: Wikipedia, The Free Encyclopedia. URL: http://en.wikipedia.org/wiki/List%5C_of%5C_most%5C_expensive%5C_paintings (visited on 05/20/2014).